



University
POLITEHNICA
of Bucharest



Faculty of
Automatic Control
and Computers



Computer Science
and Engineering
Department

Diagram Generation Using Genetic Algorithms and Orthogonal Routing

13th RoEduNet Conference – September 2014

Author

Andrei-Alexandru Musat

Andrei Voinescu



Introduction

- Diagrams are used as a tool to represent data and concepts
- Frequently utilised in software and hardware development
- Computing a proper drawing is a difficult task



- NP-complete
- Classic algorithms have limitations
- Available software presents various drawbacks
- Difficult to satisfy every user request



Task Description

- Embedding a graph in the plane
- Ensure planarity, clarity, orthogonality
- Optimize performance
- Allow user customization

- Planar graphs have no intersecting edges
- Kuratowski's Theorem defines planar graphs
- Booth-Lueker testing algorithm which runs in linear time



Graph layout

- Traditional methods: grid layout, tree layout, force-directed placing
- Experimental approach combines ideas
- Utilizes genetic algorithms
- Result is more clear and straightforward



Edge Routing

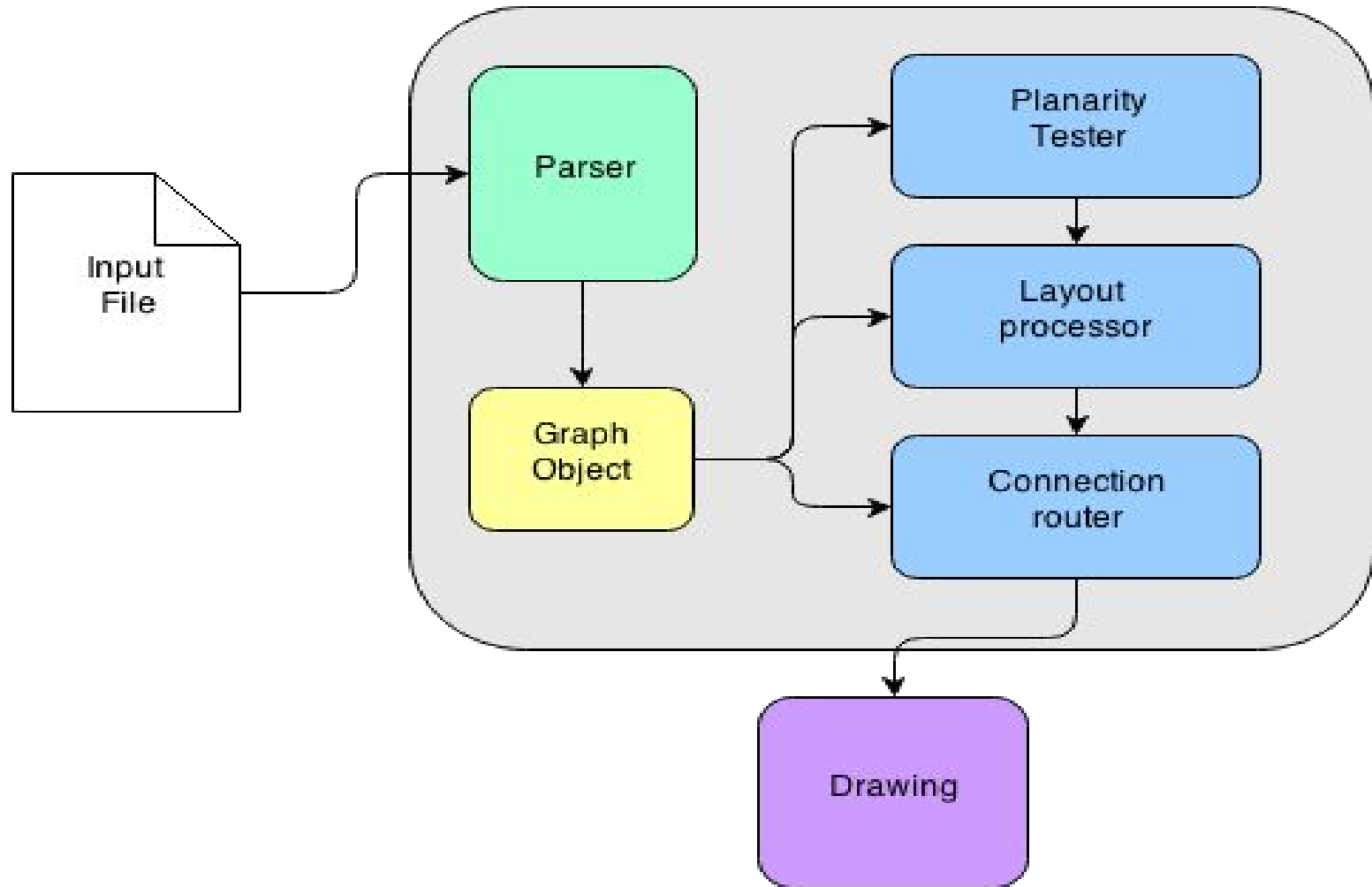
- Paths represented by orthogonal connectors
- Ensures clarity but occupies more space
- Advantageous for properly layed out graphs



- Four main modules
- Parser computes graph data from input files
- Planarity tester using Chiba-Nishizeki algorithm
- Layout processor based on genetic algorithms
- Edge router with orthogonal connectors



System Architecture





Implementation

- Modules implemented using Java language
- Eclipse Draw 2D used as graphical library
- Application integrated with the Eclipse IDE

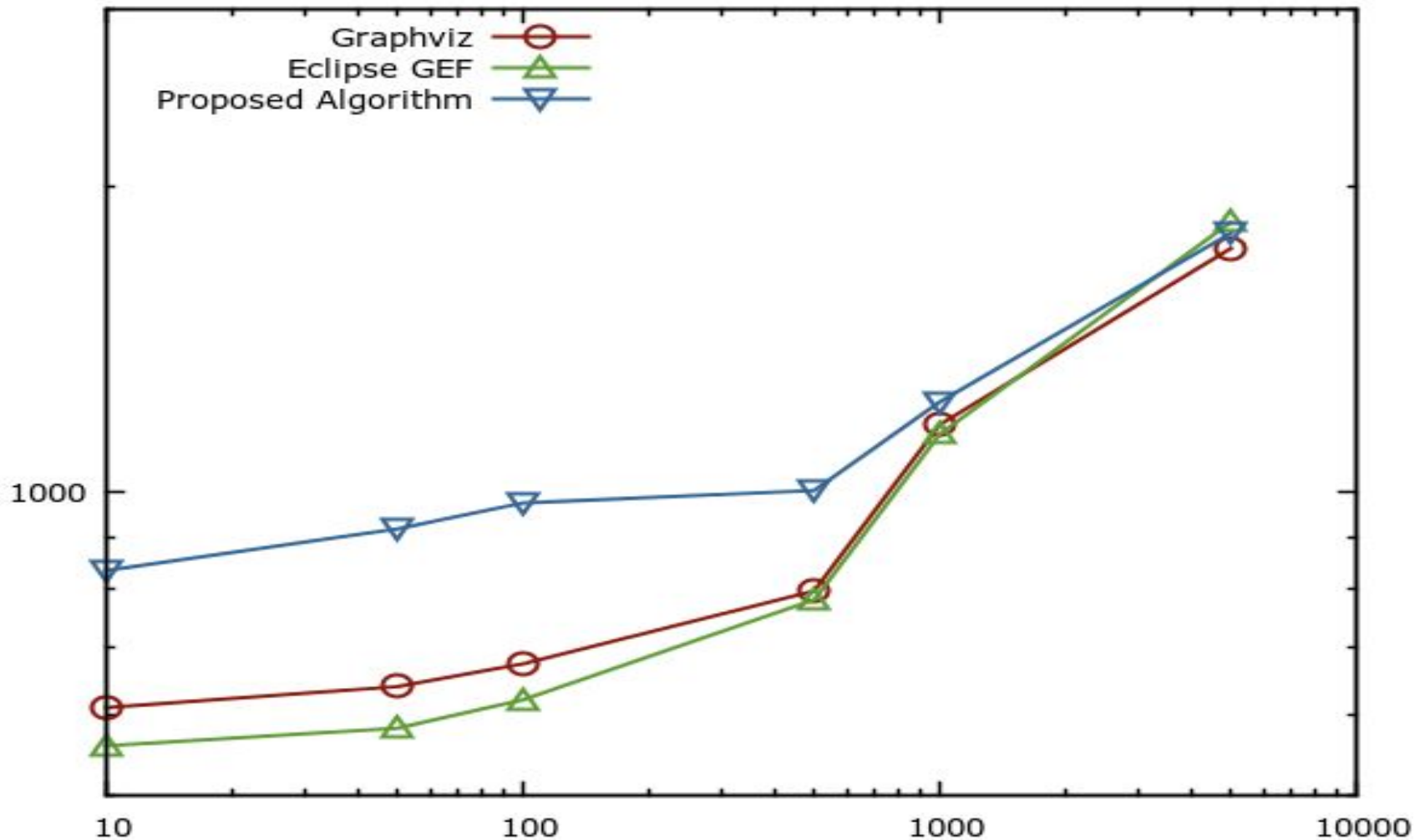


Results - Benchmarking

- Benchmarking for main processing modules
- Available open-source software: Graphviz dot and Eclipse Gef used as reference

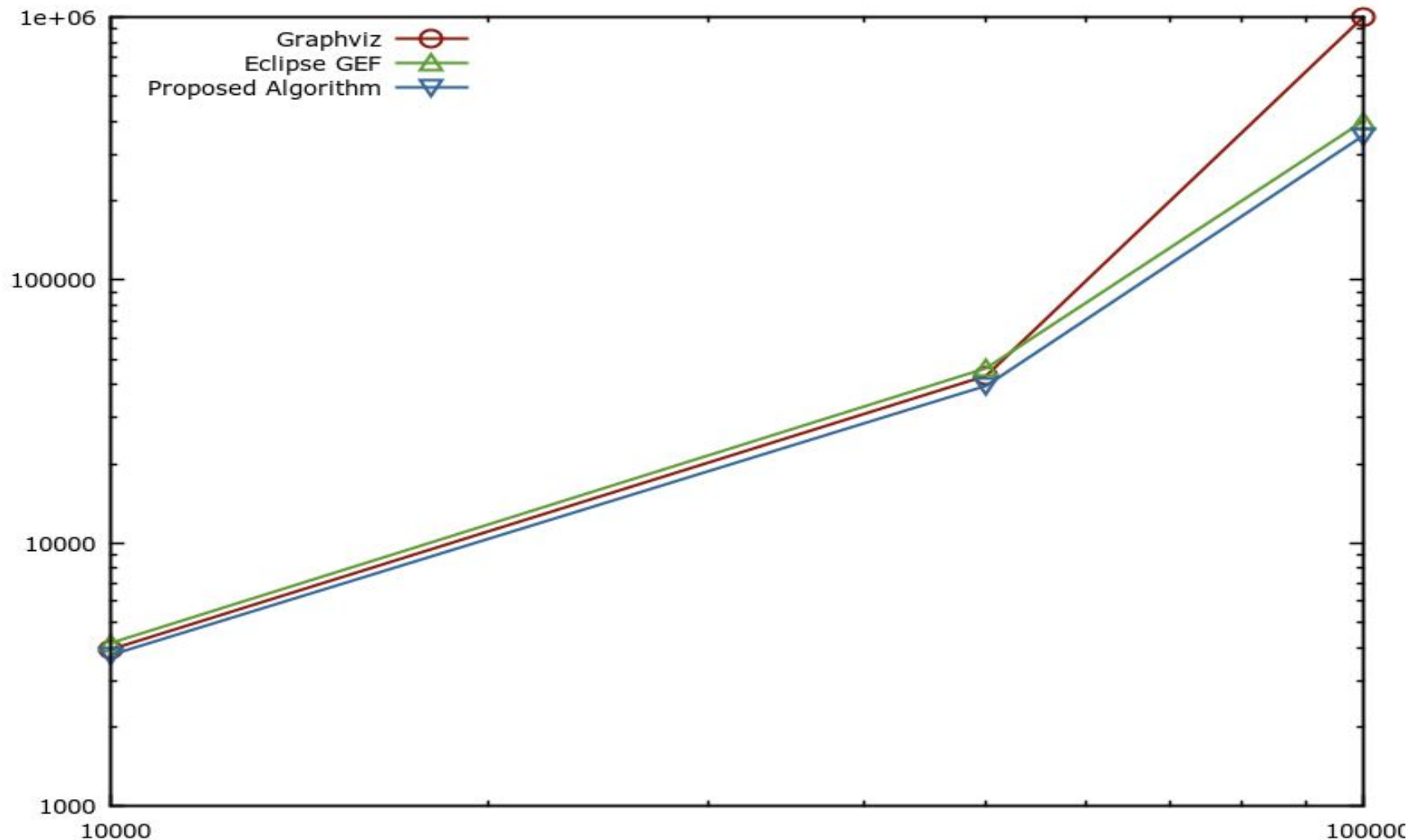


Benchmark – Small Graphs





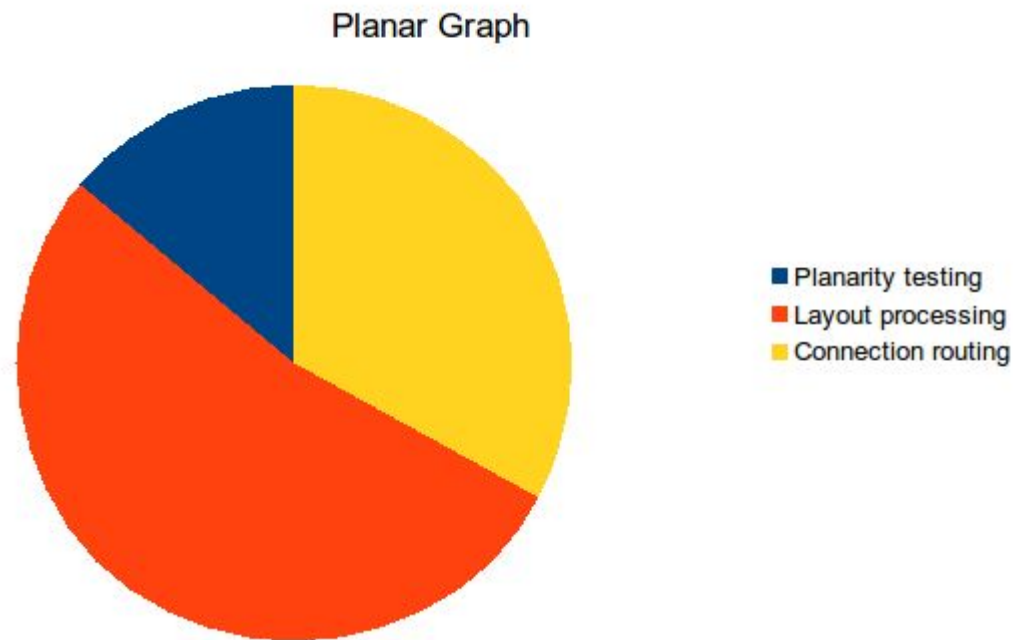
Benchmark – Large Graphs





Performance Testing

- Identify intensively used module
- Possibility for optimization





User Oriented Features

- Interact with final drawing
- Modify position of nodes
- On the fly routing for obstructed paths
- Pin elements



Conclusions

- Modern approach to a classic problem
- Using genetic algorithms improves performance
- Modularisation favours optimization
- Manual user modifications of computed diagram provides insight for further improvements



Questions

- Thank you for your time and attention!

